**Visual modelling**

I’ve been interested in modelling some well-known visual phenomena.

Strictly in 2D.

Partly because I’m intrigued by the visual algorithms applied by our perception in identifying the elements of our familiar world.

Let me give you an example of what I mean.

Seeing a tree, even from a distance, you immediately recognize it, of course, partly because of some contextual information, but – I believe --, primarily because we all possess the means to quasi-automatically analyze and identify in a split second the visual constellation corresponding to a tree.

The challenge, then, was to construct an algorithmic image of a tree, using random parameters so that it produces different images each time it runs.

The process of designing the algorithm obviously involves a rather arbitrary series of decisions, as to the using of abstraction, simplification, generalization on one side and a fine sense for the specific details on the other.

In this case I set out to model a tree as a trunk and a collection of branches. A barren tree, so to speak, without leaves of roots.

My choice was to assume, that a trunk with offshoots branches out alongside is the single defining constituent of visual structure. With each branch and twigs showing the same structure as the trunk plus the primary branches (apart from its actual random parameters). This obviously defines a model of a recursive or fractal character. A tree basically consisting of smaller trees branching out from the trunk.

This, of course, may not be a very prices description of an actual tree, whose twigs are probably more intricately tangled than the of the trunk or of the main branches.

Yet I made the arbitrary (if not the least conscious) decision to take this level of truthfulness for my model.

So far, so good.

The task, then, was to draw a trunk, with randomly chosen points along it as the starting positions of further branches.

*# this produced the images, only this time with recursive function calls*

from PIL import Image, ImageDraw  
import numpy as np  
import random  
  
im = Image.new(**'L'**, (2000, 1600))  
  
draw = ImageDraw.Draw(im)  
w = 1800  
h = 1600  
n = 10  
q = 0.88 *# parameter 1: section contraction rate*random.seed()  
startx = w // 2  
starty = h - 1  
startlength = 80.0 *# parameter 2. starting section*startangle = 0.5 \* np.pi  
  
def branch(actualx, actualy, actuallength, actualangle):

*# draws a branch recursively* x = actualx  
 y = actualy  
 l = actuallength  
 angle = actualangle *# section angle changing in second order* angle2 = 0.0 *# amount of section angle change* while (l > 4.0):  
 *# length of minimal section – with 2.0 rather fuzzy, with 10 clean  
 # parameter 3: minimal section* u = int(x + l \* np.cos(angle))  
 v = int(y - l \* np.sin(angle))  
 l = int((q + 0.06 \* random.randint(0,2)) \* l)

*# parameter 4: range for section contraction rate.*

*# here: q = 0.88...0.94* curvature = 0.000035 \* (random.randint(0, 1000) - 501) \* np.pi  
 *# quasi curvature: change of angle change*

*# rather sensitive. actual value: = 0.0175 Pi = 3.15 degrees  
 # parameter 5: section curvature, actually 0.0175 Pi* angle2 += curvature  
 angle += angle2  
 draw.line(((x, y, u, v)), fill=255, width= int(0.4 \* l), joint=**"curved"**)  
 x = u  
 y = v  
 if (random.randint(0, 100) < 27) and (random.randint(0, 45) > np.log(l)):  
 *# parameter 6: branching probability: here 0.27 AND (0.9... 1,0))*

*# a strange condition  
 # parameter 7: branching angle, actual value = - Pi/4 ... Pi/4* branch(x, y, l, angle + .025 \* (random.randint(0, 20) - 10) \* np.pi)  
 return()  
  
branch(startx, starty, startlength, startangle)  
im.show()  
im.save(**"randagasfa1recursive1.jpg"**)